Knihovna pro určení vzájemně podobných fotografií vhodného pro produkční provoz

Bc. Dobroslav Pelc

Diplomová práce
2018

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky

*** Nascanované zadání, strana 1 ***

*** Nascanované zadání, strana 2 ***

Prohlašuji, že

- beru na vědomí, že odevzdáním diplomové práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb. o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších právních předpisů, bez ohledu na výsledek obhajoby;
- beru na vědomí, že diplomové práce bude uložena v elektronické podobě v univerzitním informačním systému dostupná k prezenčnímu nahlédnutí, že jeden výtisk diplomové práce bude uložen v příruční knihovně Fakulty aplikované informatiky. Univerzity Tomáše Bati ve Zlíně a jeden výtisk bude uložen u vedoucího práce;
- byl/a jsem seznámen/a s tím, že na moji diplomovou práci se plně vztahuje zákon č. 121/2000 Sb. o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších právních předpisů, zejm. § 35 odst. 3;
- beru na vědomí, že podle § 60 odst. 1 autorského zákona má UTB ve Zlíně právo na uzavření licenční smlouvy o užití školního díla v rozsahu § 12 odst. 4 autorského zákona;
- beru na vědomí, že podle § 60 odst. 2 a 3 autorského zákona mohu užít své dílo – diplomovou práci nebo poskytnout licenci k jejímu využití jen připouští-li tak licenční smlouva uzavřená mezi mnou a Univerzitou Tomáše Bati ve Zlíně s tím, že vyrovnání případného přiměřeného příspěvku na úhradu nákladů, které byly Univerzitou Tomáše Bati ve Zlíně na vytvoření díla vynaloženy (až do jejich skutečné výše) bude rovněž předmětem této licenční smlouvy;
- beru na vědomí, že pokud bylo k vypracování diplomové práce využito softwaru poskytnutého Univerzitou Tomáše Bati ve Zlíně nebo jinými subjekty pouze ke studijním a výzkumným účelům (tedy pouze k nekomerčnímu využití), nelze výsledky diplomové práce využít ke komerčním účelům;
- beru na vědomí, že pokud je výstupem diplomové práce jakýkoliv softwarový produkt, považují se za součást práce rovněž i zdrojové kódy, popř. soubory, ze kterých se projekt skládá. Neodevzdání této součásti může být důvodem k neobhájení práce.

Prohlašuji,

- že jsem na diplomové práci pracoval samostatně a použitou literaturu jsem citoval. V případě publikace výsledků budu uveden jako spoluautor.
- že odevzdaná verze diplomové práce a verze elektronická nahraná do IS/STAG jsou totožné.

Ve Zlíně .....

podpis autora

ABSTRAKT

Cílem této práce je analýza možností pro určení vzájemně podobných fotografií. Na základě analýzy student vybere nejvhodnější návrh řešení pro potřeby reálného produkčního provozu. Výslednou komponentu realizuje formou distribuované služby.

Klíčová slova: Přehled klíčových slov

ABSTRACT

Text of the abstract

Keywords: Some keywords

Zde je místo pro případné poděkování, motto, úryvky knih, básní atp.

OBSAH

ÚVOD

První odstavec pod nadpisem se neodsazuje, ostatní ano (pouze první řádek, odsazení vertikální mezy odstavci je typycké pro anglickou sazbu; czech babel toto respektuje, netřeba do textu přidávat jakékoliv explicitní formátování, viz ukázka sazby tohoto textu s následujícím odstavcem).

Formátování druhého odstavce. Text text text text text text text text text text text text.

I. TEORETICKÁ ČÁST

1 Klasifikace řešených oblastí

Pro porovnání dvou fotografií a jejich vzájemné vyhodnocení jako podobné či nikoliv bude zaveden koeficient podobnosti fotografií (dále jen KoP). Postupně budou popsány všechny silné i slabé stránky pro možnosti výpočtu KoP. Základní klasifikací je rozdělit si rozdíly mezi vzorovou a referenční fotografií na konstruktivní a destruktivní změny.

1.1 Konstruktivní změny fotografie

Konstruktivní změny fotografie jsou takové změny, které jsou viditelné pro lidské oko, avšak nemění zásadně charakter obrázku pro strojové zpracování. Možné změny (případně jejich kombinace) jsou rozvedeny níže.

1.1.1 Návrh řešení

Pro porovnání dvou podobných fotografií, které obsahují pouze konstruktivní změny, plně postačí křížová korelace, která bude urychlena (v rámci optimalizace HW času) pomocí diskrétní Fourierovy transformace [10].

1.1.2 Oblast zájmu

Touto numerickou metodou lze detekovat většinu nejčastějších záměrných modifikací fotografií (např. automatické doostření a následné uložení jako kopie). Lze sem zahrnout jak změny vlastností vázané na původní fotografii, tak drobné změny v původním obsahu fotografie.

Změny vlastností

- Změna sytosti barev
- Změna kontrastu
- Změna jasu

Změny obsahu

- Vodotisk
- Logo
- Šum

1.2 Destruktivní změny fotografie

Destruktivní změny fotografie, jsou viditelné pro lidské oko, ale již lidským okem je často problém tyto fotografie bezpečně prohlásit za podobné. Ještě hůře je na tom strojové zpracování, pro které se výrazně mění charakter porovnávaných fotografií.

1.2.1 Návrh řešení

Výpočet Hausdorfovy vzdálenosti mezi konvexními polyedry, které reprezentující hrany v obrazu [15].

1.2.2 Oblast zájmu

- Změna komprese (rozmazaná fotografie)
- Změny rozlišení
  - Ořez (v jedné nebo obou dimenzích
  - Deformace (v jedné nebo obou dimenzích)

1.3 Kombinace konstruktivních a destruktivních změn

Kombinace konstruktivních a destruktivních změn je vždy potřeba vyhodnotit nad konkrétním případem. Platí také, že jsou velmi obtížně řešitelné. V závislost na míře změn lze často rozpoznat stejně jako čistě konstruktivní změny.

1.3.1 Návrh řešení

Redukce fotografie na její prahovou velikost jako příprava na křížovou korelaci viz konstruktivní změny [14].

1.3.2 Oblast zájmu

- Asymetrická změna obou stran s čímkoliv
- Změna kvality v důsledku zhoršeni komprese s čímkoliv
- Logo nebo vodotisk v kombinaci s předcházejícími

1.4 Výběr reprezentativního vzorku

Pro skupinu vzájemně si podobných fotografií vybereme nejvhodnějšího kandidáta, který bude následně ostatní fotografie zastupovat. Jde o experimentální postup.

1.4.1 Návrh řešení

Bude zaveden koeficient zastupitelnosti (dále jen KoZ), který je zjednodušeně určen jako $VELIKOST * OSTROST + JAS$. Přičemž platí, že vyšší hodnota koeficientu zastupitelnosti znamená kvalitnější fotografii (nikoliv na oko hezčí fotografii). Reprezentativní vzorek bude fotografie, která bude vybraná ze skupiny podobných fotografií na základě KoP, s nejvyšším KoZ.

1.4.2 Oblast zájmu

Střední hodnota jasu je opět určena pomocí světlostní matice. Pouze zohlednění zda fotka není příliš jasná nebo příliš tmavá. Jde o jednoduchý algoritmus. Na výsledek nemá zásadní vliv.

Poměrný počet hran slouží jako test rozmazanosti fotografie. K realizaci se používá konvoluční matice (více v samostatné kapitole věnované této problematice) s vhodným jádrem typu horní i dolní propusť (s celkovým součtem 0). Na výsledek má největší dopad.

Rozlišení fotografie je bráno jako klasická velikost fotografie v px (větší $\Leftrightarrow$ lepší).

2 Koeficient podobnosti dvou fotografií

Jde o základní ukazatelem podobnosti dvou fotografií. KoP leží na intervalu $< 0, 1 >$. Přičemž hodnoty blížící se 1 symbolizují podobné fotografie. Interval podobnosti byl na základě testovacích pokusů stanoven na $< 0, 07, 1)$. Hodnota 1 znamená duplicitní fotografii. Byla z intervalu vyloučena, jelikož jsou fotografie nejprve unifikovány pomocí otisku MD5 [6]. Výpočet KoP provedeme v šesti krocích, z čehož jsou čtyři kroky přípravné (optimalizační) a pouze dva kroky reálně ovlivňují výsledný KoP.

1. Změna velikosti fotografie
2. Převod na světlostní matici
3. Převod do vlnového spektra
4. Křížová korelace (Cross correlation method)
5. Převod zpět z vlnového spektra
6. Výpočet KoP z výsledné matice

Body 1 - 3 slouží jako přípravné a aplikují se na obě fotografie (vzorová a referenční). Do bodu 4 tedy vstupují dvě matice (pro každou fotografii jedna). Výstupem 4. bodu je již jen jedna matice, která je v bodě 6 vyhodnocena do výsledného KoP.

2.1 Změna velikosti fotografie

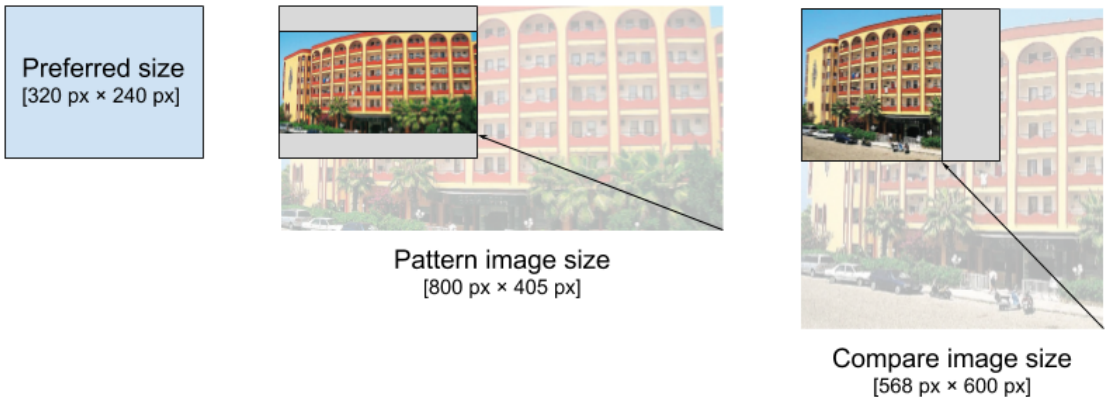
Jedná se o nezbytný přípravný krok, jehož cílem sjednotit u obou fotografií počet bodů a tím pádem také počet prvků v maticích, které vzniknou v následujícím kroku výpočtu KoP. Jako referenční velikost byla stanovena

- Šířka: 320 px,
- Výška: 240 px,

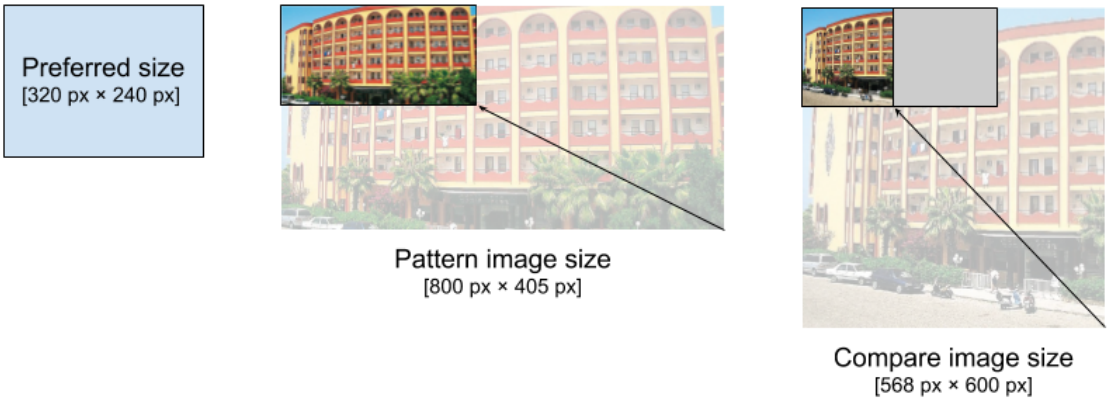
které vyšli jako nejvhodnější kompromis mezi relevancí výsledku a HW časem nutným k jeho zpracování.

Změna velikosti vzorové fotografie na [320×?] nebo [? × 240] se provádí v závislosti delší straně. Kratší strana je dopočítána podle původního poměru stran. Výsledný rozměr není doplněn nulami na plnou referenční velikost [320 × 240]. Vzorová fotografie tedy určuje velikost, na kterou musíme upravit referenční fotografii. Pokud má referenční fotografie jiný poměr stran, bude doplněna nulami, aby nevznikaly prázdná místa.

Takto připravené fotografie nejsou při dalším zpracování komutativní, pokud mají vzájemně jiný poměr stran. V důsledku to znamená, že se musí porovnat fotografie v obou směrech (jak vzorová vůči referenční, tak referenční vůči vzorové). Můžete si to ukázat na příkladu (Obr. 2.1) a (Obr. 2.2).



Obr. 2.1 Statický scaling obrázků (komutativní)



Obr. 2.2 Dynamický scaling obrázků (diskomutativní)

Přesto, že je tento proces HW dražší, než jeho komutativní varianta, získáme díky tomu řádově lepší relevanci výsledků (zejména pokud je jedena ze dvou fotografií velmi nekvalitní, případně má nižší nativní rozlišení, než je referenční).

2.2 Převod na světlostní matici

Světlostní matice je fotografie co by rastrová bitmapa převedená do dvourozměrné matice (někdy též šedotónový obraz; v anglicky psaných textech k nalezení pod názvem brightness-matrix). Formálně je to dvourozměrná diskrétní veličina, reprezentovaná maticí druhého řádu [7]. Každý bod původní fotografie - pixel [8] (dále jen px) je z RGB [9] hodnot převeden na hodnotu intenzity jasové funkce.

Původní px je reprezentován jako tří prvkové pole s hodnotami na intervalu $< 0, 255 >$.

- R => red,
- G => green,

- B => blue,

Výsledná hodnota intenzity jasové funkce px se spočítá jako střední hodnota z hodnot jednotlivých složek px (R, G a B). Jak napovídá interval možných hodnot, jeden px převedený na prvek světlostní matice zabírá v operační paměti 1 Byte (1 Byte = 8 bit => osmibitová barevná hloubka => 256 stupňů šedi). Na jedenu plnou referenční fotografii je tedy potřeba 75 KB.

2.3 Převod do–z vlnového spektra

Převodem do vlnového spektra dosáhneme zajímavé výkonností optimalizace [10]. Kdybychom tento krok z celého procesu výpočtu KoP vynechali (stejně jako následně nezbytný převod zpět z vlnového spektra), museli bychom udělat křížovou korelaci v normálním spektru. Tzn. $O(n^4)$ operací, kde n je velikost strany čtvercové matice.

Pokud ale nejprve provedeme Diskrétní Fourierovu transformaci (DFT) pro převod do vlnového spektra, až následně křížovou korelaci a nakonec Zpětnou Fourierovu transformaci (IFT), ušetříme jeden řád hodnosti počtu operací. Budeme potřebovat $O(n^3 * logn)$ operací. To je za předpokladu matice reprezentující 200 $\times$ 200 px fotografii rozdíl dvou řádů operací.

- Klasické spektrum: $200^4 = 1.6 * 10^9$ operací
- Vlnové spektrum: $200^3 * log200 = 6.1 * 10^7$ operací

2.4 Křížová korelace

Korelace je nejdůležitější krok celého výpočtu KoP. Umožní pomocí jednoduchých matematických operací rozhodnout, zda jsou světlostní matice vzorové a referenční fotografie podobné. Míra podobnosti je vyjádřena hodnotami korelačních koeficientů sestavených do jedné matice (zatím ještě ve vlnovém spektru). Čím více se výsledné hodnoty blíží 1, tím více si jsou fotografie podobné v daném bodě.

Celý proces není citlivý na konstruktivní změny. Dokáže tedy podobnost vyhodnotit bez ohledu na změnu sytosti barev, kontrastu či jasu. Stejně tak výsledek není ovlivněn přidáním vodotisku nebo šumem ve fotografii. Přidání loga do fotografie již sice sníží KoP, ale pokud není logo přes polovinu fotografie, je stále bezpečně rozpoznána jako podobná či nikoliv.

Formálně je korelace zejména statistický pojem, který označuje vzájemný lineární vztah mezi znaky nebo veličinami. Míra korelace je daná korelačním koeficientem [11]. Vzorec pro výpočet: $\rho_{A,B} = \dfrac{(A - \mu_A) \times (B - \mu_B)}{\sigma_A \sigma_B}$

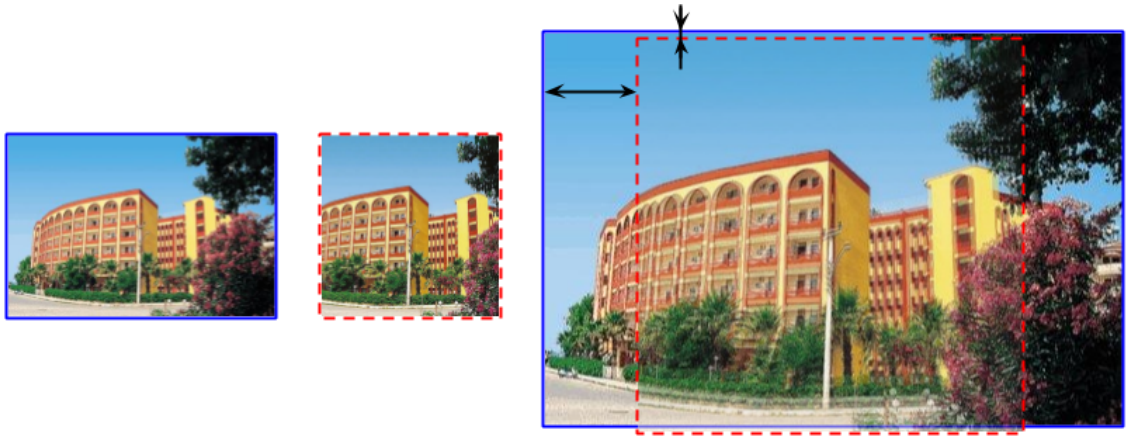
Na první pohled se může zdát složitý, ale skrývá v sobě tři jednoduché ale důležité kroky.

1. K maticím A a B se spočítá rozptyl σ_A, σ_B a střední hodnota μ_A, μ_B.
2. Odečtením střední hodnoty od matice $A - \mu_A$, $B - \mu_B$ je dosaženo **invariace nastavení jasu**.
3. Dělením rozptyly $\sigma_A \sigma_B$ je zajištěna **invariace nastavení kontrastu**.

Za předpokladu, že objekty na porovnávaných fotografiích nejsou vůči sobě v prostoru posunuty, poskytuje korelační matice odpověď na otázku, zda jsou si dvě fotografie podobné. V praxi ale tento model příliš nenastává. Naopak je velmi časné, že je porovnáván např. výřez z fotografie oproti originálu, případně posunuté fotografie po horizontální či vertikální ose.

Tuto problematiku řeší křížová korelace (Cross correlation method) [12]. Princip samotné korelace je stejný, pouze se opakuje s částečným posunem tak, aby pokryla všechny možné kombinace mezi dvěma fotografiemi. Příklad je asi nejlépe vidět na původních fotografiích, jak ukazuje (Obr. 2.3) a (Obr. 2.4).

Obr. 2.3 Příklad proložení dvou fotografií s použitím klasické korelace



Obr. 2.4 Příklad proložení dvou fotografií s použitím křížové korelace

2.5 Výpočet výsledného koeficientu

Po převodu z vlnového spektra zpět, vychází již matice z které jsou vyloučeny imaginární hodnoty a pracujeme tedy jen s reálnými čísly. Výsledný KoP pro dané fotografie je dám nejvyšším nalezeným korelačním koeficientem v matici.

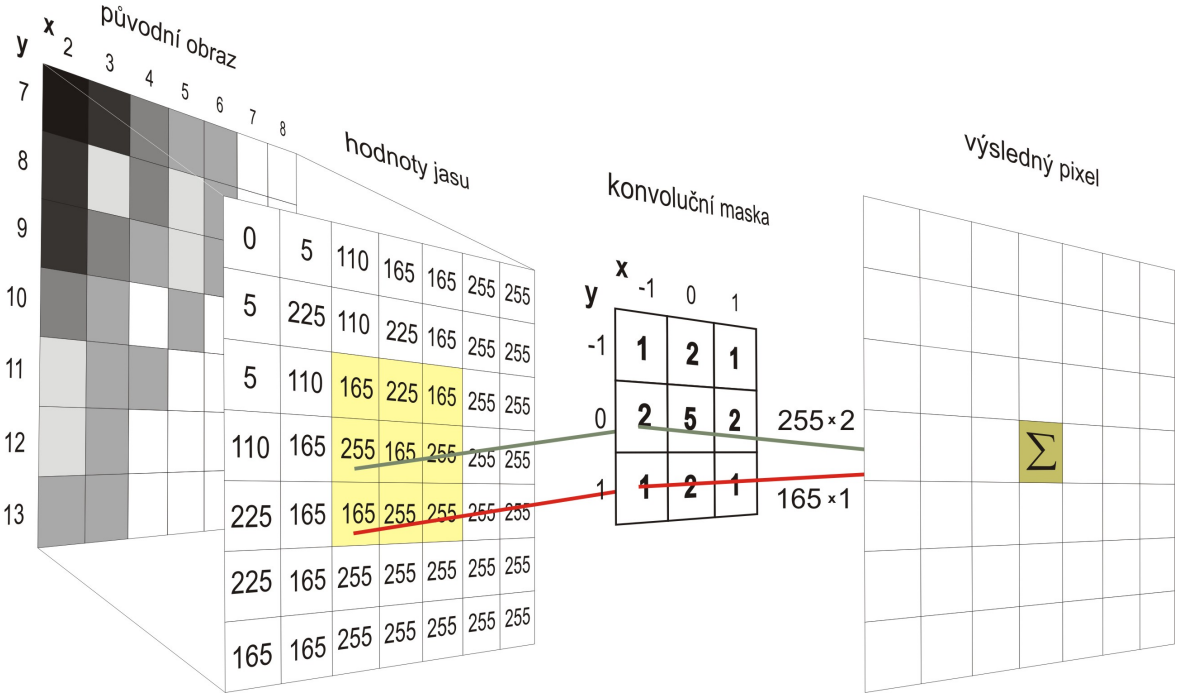
3 Koeficient zastupitelnosti dvou fotografií

Určení KoZ je experimentální proces založený na opakované aplikaci konvoluční matice s různým jádrem (konvoluční maskou). Bude jí věnována větší pozornost v projektové části. Zde budou představeny pouze techniky nutné k jejímu dosažení. Jelikož převod na světlostní matici a převod z–do vlnového spektra byl již představeny, nebudou zde již znovu uvedeny.

3.1 Diskrétní 2D konvoluce

Konvoluce [19] je matematický operátor zpracovávající dvě funkce. V algoritmech zpracovávající dvourozměrný diskrétní obraz (např. v počítačové grafice) má konvoluce následující tvar: $(f * h)(x, y) = \sum_{i=-k}^{k} \sum_{i=-k}^{k} f(x - i, y - j) \bullet h(i, j)$

Názorná ukázka konvoluce je vidět na přiloženém obrázku (Obr. 3.1).



Obr. 3.1 Příklad Diskrétní 2D konvoluce [19]

3.2 Konvoluce a volba jádra

Konvoluce ve světě počítačové grafiky je operace s obrázkem (co by matice) pomocí (jiné) matice zvané „jádro“ (nebo též konvoluční maska). Jako první matice se používá obrázek určený k úpravě. Obrázek je vlastně dvojrozměrná pravoúhlá souřadnicová síť pixelů. Použité jádro závisí na požadovaném efektu.

V našem případě je požadovaný na filtr detekce hran [20]. Základní jádra pro detekci hran mají součet roven nule (Tab. 3.1 a Tab. 3.2). Nejsou vhodné na obrázky zatížené šumem. Ten je nezbytné před aplikací těchto filtrů eliminovat.

Tab. 3.1 Jádro pro detekci hran (horizontálně a vertikálně)

0	1	0
1	-4	1
0	1	0

Tab. 3.2 Jádro pro detekci hran (horizontálně, vertikálně a šikmé hrany)

1	1	1
1	-8	1
1	1	1

II. ANALYTICKÁ ČÁST

4 Brainstorming

Analytická metoda sloužící ke sběru myšlenek, námětů a případné zevrubné konstruktivní kritice dané problematiky. V rámci této práce byla použita v akademickém a profesním kruhu pro identifikaci základních ukazatelů pro další kroky analýzy.

4.1 Akademický kruh

Diskutovány zejména technické možnosti. Kde a jak lze vůbec porovnání fotografií provádět strojově. K další analýze vyly vyb

4.2 Profesní kruh

V kruhu s provozovatelem byly kladeny nejvyšší nároky na flexibilitu a propustnost celého řešení.

5 Diferenční analýza (GAP analýza)

5.1 Popis současného stavu

Je požadováno porovnání rastrových bitmap za účelem identifikace vzájemně podobných fotografií. Řešení je hledáno pro produkční provoz. Konzumentem cílového řešení je webový portál dovolena.cz, který má přibližně dva miliony fotografií. Průměrný počet přístupů k některé fotografii je přibližně 100 přístupů za sekundu. Obsahem fotografií jsou především o hotely a jejich okolí. Webový portál slouží spíše jako datový konsolidátor. Nabídka portálu značně ovlivňuje cílové portfolio fotografií. Podle testovacích měření se za jeden týden obmění cca 10% fotografií z celkového množství. Jako testovací vzorek byl vybrán jeden nejmenovaný hotel a jeho 35 fotografií. Zákazník webového portálu vidí všechny fotografie v nesetříděné galerii. Některé fotografie jsou unikátní, ale většina si je velmi podobná. U některých dokonce nejsou lidským okem patrné rozdíly.

5.2 Popis cílového stavu

Konsolidované fotografie prezentované klientovi budou v maximální možné míře obsahovat unikátní fotografie. Vzájemně si podobné fotografie budou odfiltrovány a zůstane pouze jedna a to fotografie s nejvyšším KoZ. Klient nebude čekat na zpracování podobnosti obrázků. Buď budou zpracované, pak klient uvidí jen unikáty, nebo ne a pak uvidí vše v původním stavu. V takovém případě se poměrově zvýší priorita na výpočet podobnosti fotografií tohoto hotelu vůči ostatním ve frontě na výpočet. Cílové řešení musí být schopno operovat řádově s jednotkami milionů fotografií s týdenní fluktuací 15%.

5.2.1 Nefunkční požadavky

- Bezúdržbový systém
- Nevyžadující v průběhu času další financování
- Minimální vstupní investice
- Maximální kompatibilita s aktuálním HW
- Programovací jazyk Java [24]

5.3 Rozdíly

- Nově vznikne nástroj pro určení KoP.
- Nově vznikne nástroj pro určení KoZ.
- Fotografie jednoho hotelu budou oindexovány a vnitřně škálovány do skupin pomocí koeficientů výše.
- Dojde k navýšení celkového počtu fotografií.
- Zvýší se fluktuace fotografií (na očekávaných 15%).

5.4 Návrh variant k dosažení cíle

Pilířem celého řešení bude backendová strana cílového konzumenta. Limity a také jednotlivé možnosti pro realizaci jsou velmi omezeny nutností integrovat do současného řešení. Nejen z těchto důvodů má serverová strana spíše podpůrný charakter v projektu jako celku. Její význam je spíše v propojení všech jednotlivých komponent. Dojde tedy k modifikaci existujícího produkčního server-side prostředí. Nově zde poběží služba, která bude

- poskytovat zadání na určení KoP,
- poskytovat metadata nezbytná pro distribuci výpočtu,
- konzumovat výsledek distribuované operace,
- kompletovat zpracovaná data do cache vhodné pro silný organický provoz.

Naopak klientská strana je naprosto autonomní. Pro realizaci lze použít jak libovolnou platformu, tak libovolné technologie. Jediným technickým limitem je schopnost standardizovaným způsobem komunikovat se serverovou stranou pomocí SOAP api [21].

5.4.1 Výpočet KoP a KoZ na CPU produkčního serveru

Základní myšlenka je využít nejdostupnější produkční HW a na zavedeném serveru spustit novou službu. Hlavní výhodou je dostupnost produkčního HW ve vlastním datacentru cílového konzumenta. Podstatnou nevýhodou fakt, že pro určení koeficientů výše není CPU [22] ideální platforma.

5.4.2 Výpočet KoP a KoZ na CPU produkčního serveru s paralelizací na GPU

Základní myšlenka je osadit do produkčního serveru dedikovanou pracovní grafickou kartu a vybrané části procesu výpočtu KoP a KoZ optimalizovat pro zpracování na GPU [23].

5.4.3 Výpočet koeficientů na PC farmě

Základní myšlenka je využít HW osobních PC, kterých je v každé větší firmě požehnaně a neprovádět výpočet na jednom stroji, na každém dostupném stroji.

5.5 Zhodnocení variant

Jako nejvýhodnější varianta pro realizaci vychází PC farma. Jako záložní řešení, lze využít produkční server s výpočtem na CPU. Pokud bude zvolen vhodný programovací jazyk (např. požadovaný programovací jazyk Java [24]), bude výsledné řešení přenositelné.

Zhodnocení výkonových rozdílů vychází z benchmarkingu ukazuje tabulka (Tab. 5.1), který byl vyhodnocen jako pomocná analýza níže.

Tab. 5.1 Tabulka výsledků jednotlivých variant krátkodobého testu

Varianta	Týdenní přírůstek	Všechny fotografie	Teoretická rezerva
CPU	6 dní	27 týdnů	3,7%
GPU	8 hodin	2 dny 2 hodiny	300%
PC farma (300 ks)	2,5 hodin	16 hodin	700%

Zhodnocení investičních rozdílů ukazuje tabulka (Tab. 5.2). Vychází z předpokladu, že x je investiční konstanta v Kč, vztažená na cenu jednoho produkčního serveru bez GPU. Pro zjednodušení je stanovena na 100.000 Kč (není příliš daleko od reality).

Tab. 5.2 Tabulka výsledků jednotlivých variant krátkodobého testu

Varianta	Pořizovací cena [Kč]	Nutná investice [Kč]
CPU	x	0
GPU	$10x$	$10x$
PC farma (300 ks)	$30x$	0

5.5.1 Výpočet KoP a KoZ na CPU produkčního serveru

Výsledky banchmarkingu

- Za necelých 6 dní spočítá týdenní přírůstek.
- Za zbylou dobu z týdenního cyklu dále vypočítá přibližně 3,7% z celkového objemu sto milionu koeficientů.
- Pro plné vypočítání všech koeficientů potřebuje dalších cca 27 týdnů.
- Rezerva pro další růst (navýšení celkového počtu fotografií) je cca 18%.

5.5.2 Výpočet KoP a KoZ na CPU produkčního serveru s paralelizací na GPU

Pro realizaci tohoto scénáře je nezbytná vysoká vstupní investice. Běžně dostupný produkční server nedisponuje GPU. Pořizovací cena takového serveru je řádově vyšší. Dále je nutné připočíst cenu vlastní dedikované grafické karty, která cenově převyšuje cenu serveru. Její výběr je velmi omezen kompatibilitou s produkčními servery. Zástupce firmy Dell byl schopný nacenit na vlastní server pouze dvě karty s možností garancí non-stop provozu a výměnou do 24 hodin.

Výsledky banchmarkingu

- Za necelých 8 hodin spočítá týdenní přírůstek.

- Za část ze zbylé doby týdenního cyklu dále vypočítá 100% z celkového objemu sto milionu koeficientů.
- Pro plné vypočítání všech koeficientů potřebuje celkem 2 dny.
- Rezerva pro další růst (navýšení celkového počtu fotografií) je cca 300%.

5.5.3 Výpočet koeficientů na PC farmě

Výsledek jednoho kancelářského PC je zanedbatelný a nemůže se rovnat předchozím variantám. Vezmeme-li v úvahu, že těchto strojů je k dispozici 16 hodin denně více než 300 kusů, začíná to již vypadat v číslech jinak.

Výsledky banchmarkingu

- Za 2,5 hodiny spočítá týdenní přírůstek (za předpokladu 300 aktivních PC).
- Za část ze zbylé doby týdenního cyklu dále vypočítá 100% z celkového objemu sto milionu koeficientů.
- Pro plné vypočítání všech koeficientů potřebuje celkem 16 hodin, což je v tomto případě 1 den.
- Rezerva pro další růst (navýšení celkového počtu fotografií) je cca 700%.

Samozřejmě s tímto nestandardním krokem je spojená také revize infrastruktury, především kvalita a šířka pásma sítě, centrální správa PC aj, které je nutné v tuto chvíli zanedbat.

6 Benchmarking

Jednotlivé kandidáty pro realizaci klientské části (zavedené v diferenční analýze) podrobíme výkonnostním a srovnávacím testům. Základní předpoklady:

- Řádově je potřeba jednorázově odbavit 100.000.000 výpočtů KoP.
- Řádově je potřeba jednorázově odbavit 2.000.000 výpočtů KoZ.
- Na týdenní bázi následně odbavovat přírůstky řádově
  - 15.000.000 výpočtů KoP,
  - 300.000 výpočtů KoZ.

6.1 Testovací prostředí

6.1.1 Použitý HW

Simulace produkčního serveru je založena na pracovní stanici Lenovo D20, která je svoji sestavou velmi blízko produkčnímu serveru. Vzhledem k tomu, že disponuje starší generací CPU, bylo do ní osazeno i odpovídají GPU, aby bylo možné výsledky aproximovat na reálný produkční HW.

- Lenovo ThinkStation D20
- Operační systém Windows 10 pro 64 bit
- CPU Intel(R) Xeon(R) X5670 @ 2.93 GHz (2×cpu, 6×core, 12×thread)
- RAM 32 GB DDR3 (1066)
- GPU NVIDIA GeForce GTX 460
- HDD SAS 10.000 ot

Simulace klasického PC pro je založena na PC Dell Optiplex 780. Obecně je to velmi rozšířený kancelářský PC. Taktéž cílový konzument disponuje více než 300 ks odpovídajících kancelářských PC. Tato simulace je omezena pouze na výpočty na CPU.

- Dell Optiplex 780
- Operační systém Windows 7 pro 32 bit
- CPU Intel Core 2 Duo E7500 2,93 GHz
- RAM 4 GB DDR3
- HDD SATA 250 GB

6.1.2 Použitý SW

V úvodní fázi projektu probíhal vývoj i testování algoritmů v Matlabu [4]. Ten vnitřně používá pro zpracování DFT a IFT knihovnu FFTW [13]. Tato knihovna má dostupnou nativní implementaci v CMake [17]. Dále má pro spoustu programovacích jazyků (včetně programovacího jazyku Java [16]) připravený wrapper [18].

Algoritmu v matlabu (výpočet KoP na CPU) banchmark-cpu.m

```
close all; clc; clear;

method = 'nearest';
%method = 'bicubic';

type = 'single';

n = 35; % pocet obrazku

%obrazky
im_gpu = cell(1, n);

h = zeros(1, n, type);
w = zeros(1, n, type);

s0 = 'hotel_images\aa';

%nahravani obrazku
tic
for i = 1:n
    if i < 10
        s = [s0,'0',int2str(i),'.tif'];
    else
        s = [s0,int2str(i),'.tif'];
    end;

    im = single(imread(s)); %nahrani obrazku a prevod

    [n1,n2,n3] = size(im);
    h(i) = n1;
    w(i) = n2;
    im_gpu{i} = im; %nahani obrazku
end;
disp('Nahravani vsech obrazku:');
toc

%prevadeni obrazku na jasove matice
for i=1:n
    im_gpu{i} = sum(im_gpu{i}, 3) ./ 3; %prevod na jasovou matici
end;
disp('Prevod vsech obrazku na jasovou matici:');
toc

log_cpu = zeros(n, n, type);

%cyklus
for i=1:n
    %ft prvniho obrazku
    ft1 = fft2(im_gpu{i});
    ft1Norm = ft1 ./ abs(ft1);

    for j=1:n
        if i==j
            continue;
        end;

        scale = min(h(i)/h(j), w(i)/w(j));
        h2 = round(scale * h(j));
        h2 = min(h2, h(i));
        w2 = round(scale * w(j));
        w2 = min(w2, w(i));

        im2Sc = imresize(im_gpu{j}, [h2, w2], method);

        %doplneni druheho obrazku nulami na jednotnou velikost
        im2 = zeros(h(i), w(i), type);
        im2(1:h2, 1:w2) = im2Sc;

        %ft druheho obrazku
        ft2 = fft2(im2);
        ft2Norm = ft2 ./ abs(ft2);

        %krizova korelace ve vlnovem spektru
        ccW = ft1Norm .* conj(ft2Norm);

        %prevod do obycejneho souradnicoveho systemu
        ccC = real(ifft2(ccW));

        %hledani maximalniho korelacniho koeficientu
        ccCMax = max(ccC);
        ccCMax = max(ccCMax);
        log_cpu(i,j) = ccCMax;
        disp(['obr1: ',int2str(i),';  obr2: ',int2str(j)]);
    end;
end;
disp(['Cyklus ',int2str(n),' obrazku:']);
time = toc

clearvars -except log_cpu;
```

Výsledný KoP odpovídá proměnné ccCMax.

Algoritmus v matlabu (výpočet KoP s paralelizací na GPU) banchmark-gpu.m

```
%setenv('NSIGHT_CUDA_DEBUGGER','1')
close all; clear; clc;

%method = '_nearest';
%method = '_biquadr';
method = '_bicubic';

type = 'single';

n = 35;

path = 'kernels\ImgScale\ImgScale\kernel_';
kernel = parallel.gpu.CUDAKernel([path, type, method, '.ptx'], [path, type, method, '.cu']);

%obrazky
im_gpu = cell(1,n);

h = zeros(1, n, type);
w = zeros(1, n, type);

```

```
s0 = 'hotel_images\aa';

%nahravani obrazku na grafiku
tic
for i = 1:n
    if i < 10
        s = [s0,'0',int2str(i),'.tif'];
    else
        s = [s0,int2str(i),'.tif'];
    end

    im = single(imread(s)); %nahrani obrazku a prevod

    [n1,n2,n3] = size(im);
    h(i) = n1;
    w(i) = n2;
    im_gpu{i} = gpuArray(im); %nahrani obrazku na grafiku
end
disp('Nahravani vsech obrazku do pameti grafiky:');
toc

%prevadeni obrazku na jasove matice
for i=1:n
    im_gpu{i} = sum(im_gpu{i}, 3) ./ 3; %prevod na jasovou matici
end;
disp('Prevod vsech obrazku na jasovou matici:');
toc

log = zeros(n, n, type, 'gpuArray');

%cyklus
for i=1:n
    %ft prvniho obrazku
    ft1 = fft2(im_gpu{i});
    ft1Norm = ft1 ./ abs(ft1);

    for j=1:n
        if i==j
            continue;
        end;

        scale = (min(h(i)/h(j), w(i)/w(j)));
        h2 = round(scale * h(j));
        h2 = min(h2, h(i));
        w2 = round(scale * w(j));
        w2 = min(w2, w(i));
        scale_h = h(j)/h2;
        scale_w = w(j)/w2;

        %im2Sc = imresize(im_gpu{j}, scale);

        im2Sc = zeros(h2, w2, type, 'gpuArray');
        kernel.GridSize = w2;
        kernel.ThreadBlockSize = h2;
        im2Sc = feval(kernel, im2Sc, im_gpu{j}, int32(h(j)), int32(w(j)), scale_h, scale_w);

        %doplneni druheho obrazku nulami na jednotnou velikost
        im2 = zeros(h(i), w(i), type, 'gpuArray');
        im2(1:h2, 1:w2) = im2Sc;

        %ft druheho obrazku
        ft2 = fft2(im2);
        ft2Norm = ft2 ./ abs(ft2);

        %krizova korelace ve frekvencnim spektru
        ccW = ft1Norm .* conj(ft2Norm);

        %prevod do obycejneho souradnicoveho systemu
        ccC = real(ifft2(ccW));

        %hledani maximalniho korelacniho koeficientu
        ccCMax = max(ccC);
        ccCMax = max(ccCMax);
        log(i,j) = ccCMax;
        disp(['obr1: ',int2str(i),';  obr2: ',int2str(j)]);
    end
end
disp(['Cyklus ',int2str(n),' obrazku:']);
log_gpu = gather(log);
toc

clearvars -except log_gpu;
```

Výsledný KoP odpovídá proměnné ccCMax.

Algoritmus v matlabu (výpočet KoZ na CPU) banchmark-koz-cpu.m

```
a = cell(1,15);
        a{7}  ='image/006780_5305124.jpg';
a{10} ='image/37342_cat008802_069_01.jpg';
a{13} ='image/37342_cat009629_110_10.jpg';
a{4}  ='image/37342_010590_11837774.jpg';
a{11} ='image/37342_cat008103_113_02.jpg';
a{1}  ='image/37342_cat009015_115_06.jpg';
a{9}  ='image/37342_cat011203_140_04.jpg';
a{6}  ='image/cat013326_136_04.jpg';
a{14} ='image/cat012556_124_06.jpg';
a{5}  ='image/37342_cat010488_142_06.jpg';
a{15} ='image/cat013842_136_04.jpg';
a{8}  ='image/lbc402_zimmer1_0912.jpg';
a{12} ='image/37342_cat009533_052_02.jpg';
a{2}  ='image/37342_cat011000_055_02.jpg';
a{3}  ='image/37342_cat011855_060_02.jpg';

close all;

startfile = 1;
endfile = 15;

best = zeros(endfile - startfile+1,4);

kernel_up = [-0.0625 -0.0625 -0.0625; -0.0625 0.5 -0.0625; -0.0625 -0.0625 -0.0625];

kernel_down = [0 0.125 0; 0.125 0.5 0.125; 0 0.125 0];

for i=startfile:endfile
```

```
    im = imread(a{i});
    [m,n] = size(im(:,:,1));

    im_br = sum(im,3)/3;
    im_ft = fft2(im_br);

    kernel_up_sized = zeros(m,n);
    kernel_up_sized(1:2,1:2) = kernel_up(2:3,2:3);
    kernel_up_sized(m,1:2) = kernel_up(1,2:3);
    kernel_up_sized(1:2,n) = kernel_up(2:3,1);
    kernel_up_sized(m,n) = kernel_up(1,1);
    kernel_up_ft = fft2(kernel_up_sized);

    im_conv1_ft = im_ft.*kernel_up_ft;
    im_conv1 = ifft2(im_conv1_ft);

    kernel_down_sized = zeros(m,n);
    kernel_down_sized(1:2,1:2) = kernel_down(2:3,2:3);
    kernel_down_sized(m,1:2) = kernel_down(1,2:3);
    kernel_down_sized(1:2,n) = kernel_down(2:3,1);
    kernel_down_sized(m,n) = kernel_down(1,1);
    kernel_down_ft = fft2(kernel_down_sized);

    im_conv2_ft = im_conv1_ft.*kernel_down_ft;
    im_conv2 = ifft2(im_conv2_ft);

    im_conv1_8 = uint8(zeros(m,n,3));
    im_conv1_8(:,:,1) = uint8(im_conv1);
    im_conv1_8(:,:,2) = uint8(im_conv1);
    im_conv1_8(:,:,3) = uint8(im_conv1);
    im_conv2_8 = uint8(zeros(m,n,3));
    im_conv2_8(:,:,1) = uint8(im_conv2);
    im_conv2_8(:,:,2) = uint8(im_conv2);
    im_conv2_8(:,:,3) = uint8(im_conv2);

%    if i==1 || i==2 % || i==9
%        figure;
%        image(im_conv1_8);
%        figure;
%        image(im_conv2_8);
%    end;

    sharpness = sum(sum(im_conv1 > 20));

    oversharpness = sum(sum(im_conv1-im_conv2 > 20));

    best(i-startfile+1,1) = i;
    best(i-startfile+1,2) = sharpness;
    best(i-startfile+1,3) = oversharpness;
    best(i-startfile+1,4) = sharpness / max(250,oversharpness-250);
end;

[sbest, sx] = sort(best(:,4),'descend');
best = best(sx,:);
```

Výpočet KoZ byl nakonec ve srovnávacích testech zanedbán. Především proto, že se nemusí počítat pro kombinace fotografií, ale právě jednou pro každou fotografii a to při jejím pořízení. Průměrná doba výpočtu KoZ je 620 ms.

6.2 Výsledky testování

Byly realizovány tři testovací scénáře.

- Výpočet čistě na CPU s maximálním využitím produkčního HW.
- Výpočet na CPU s využitím GPU pro paralelizaci vybraných procesů.
- Výpočet čistě na CPU na PC farmě.

6.2.1 Výpočet koeficientů na CPU

Výpočet na CPU byl spuštěn v pěti vláknech. Z počátku vypadal průběh velmi slibně. Bohužel při delším testu se výpočet začal značně propadat. Později se ukázalo, že hlavním důvodem je odložené uvolňování RAM. Jinými slovy dokud byla další volná RAM, výpočet jel velmi rychle. S potřebou uvolnit RAM se výpočet řádově snížil.

- Krátkodobý test
  - doba: 5 minut
  - počet opakování testu: 10
  - průměrně za sekundu: 205 výpočtů obou koeficientů

- Dlouhodobý test
  - doba: 6 hodin
  - počet opakování testu: 10
  - průměrně za sekundu: 31 výpočtů obou koeficientů
- Propad o 85%

Na obrázku níže (Obr. 6.1) je výstup z profilování výpočtu na CPU. Obsahuje poměrově zvýrazněné dlouho trvající operace vůči zbytku procesu. V rámci profilování bylo zpracováno deset tisíc iterací a doby jednotlivých částí zprůměrovány.

Obr. 6.1 Vizualizace procesu výpočtu koeficientů na CPU

Celková zátěž alokovaného HW nebyla příliš slavná. Nepodařilo se efektivně využít CPU, které dosahovalo průměrné zátěže 30% zejména kvůli neustálé realokaci operační paměti, na kterou čekalo zkrátka vše. Úvodní myšlenka využít hrubou sílu produkčního HW se ukázala jako velmi špatná.

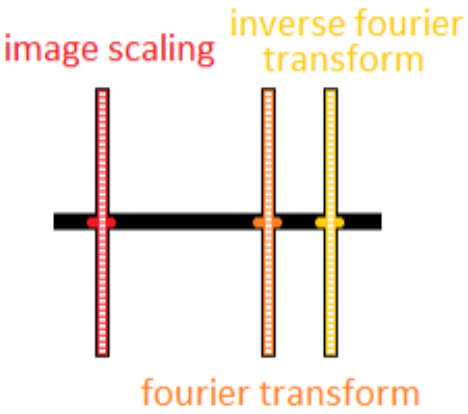
6.2.2 Výpočet koeficientů na CPU s paralelizací na GPU

Výpočet byl spuštěn v jednom vlákně na CPU. Vhodné operace pro GPU jsou delegovány pro paralelní zpracování. Nejprve bylo na GPU paralelizován pouze úvodní scaling obrázků. To nemělo příliš valný dopad na výsledky. Následně byly paralelizovány na GPU také prováděné transformace obrázků. Právě tento krok měl zásadní vliv na zrychlení celé operace. Je zde opět patrný lehký propad krátkodobého testu oproti dlouhodobému.

- Krátkodobý test
  - doba: 5 minut
  - počet opakování testu: 10
  - průměrně za sekundu: 617 výpočtů obou koeficientů
- Dlouhodobý test
  - doba: 6 hodin
  - počet opakování testu: 10
  - průměrně za sekundu: 559 výpočtů obou koeficientů
- Propad o 10%

Na obrázku níže (Obr. 6.2) je výstup z profilování výpočtu na CPU s využitím GPU pro paralelizací dlouhotrvajících operací na CPU. Obsahuje poměrově zvýrazněné

dlouho trvající operace vůči zbytku procesu. V rámci profilování bylo zpracováno deset tisíc iterací a doby jednotlivých částí zprůměrovány.



Obr. 6.2 Vizualizace procesu výpočtu koeficientů na CPU s paralelizací na GPU

Zátěž na CPU dosahuje v průměru 5%. Naopak GPU dosahuje zátěže cca 75%. Otázkou zůstává, jak dlouho je schopná GPU pracovat pod tímto permanentním zatížením.

6.2.3 Výpočet koeficientů na PC farmě

Výpočet byl spuštěn v jednom pracovním vlákně s omezenou možností alokace RAM na 1GB. Jedná se v podstatě o alternativu výpočtu obou koeficientů čistě na CPU. Stejně tak tomu odpovídal i celkový průběh zpracování, který téměř stejný, pouze s nižší dotací vypočítaných dvojic koeficientů. Lze tedy přejít rovnou na výsledky, jelikož samotné zpracování nic nového nepřineslo.

- Krátkodobý test
  - doba: 5 minut
  - počet opakování testu: 10
  - průměrně za sekundu: 10 výpočtů obou koeficientů
- Dlouhodobý test
  - doba: 6 hodin
  - počet opakování testu: 10
  - průměrně za sekundu: 6 výpočtů obou koeficientů
- Propad o 40%

Zátěž na CPU je konstantní. Jedno jádro ze dvou jede permanentně na 100%. Celková zátěž CPU tedy 50%. Vzhledem k tomu, že stoj již netrpěl na realokaci paměti, je

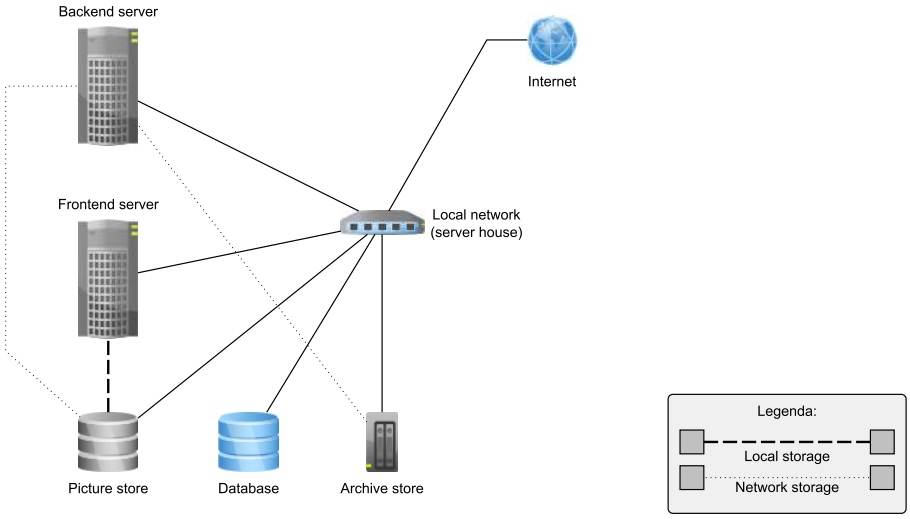
v případě PC úzkým hrdlem opravdu CPU. Ostatní sledované metriky (vytížení HDD, síťový provoz, pracovní teplota) se příliš nevychýlili z normálu.

III. PROJEKTOVÁ ČÁST

7 Příprava projektu

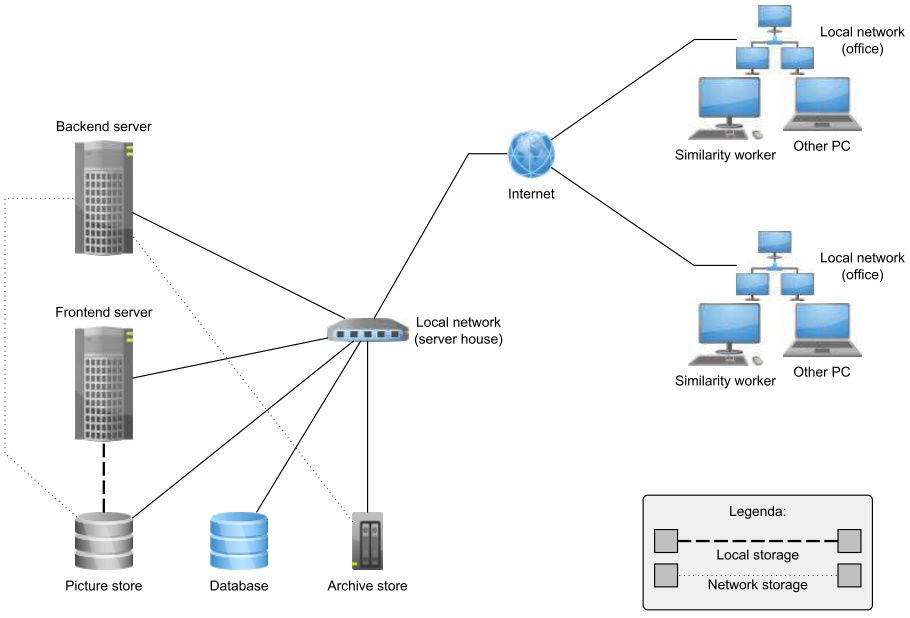
7.1 HW Architektura

7.1.1 Pohled na aktuální HW mapu



Obr. 7.1 L1 pohled aktuálního stavu

7.1.2 Pohled na plánovanou HW mapu

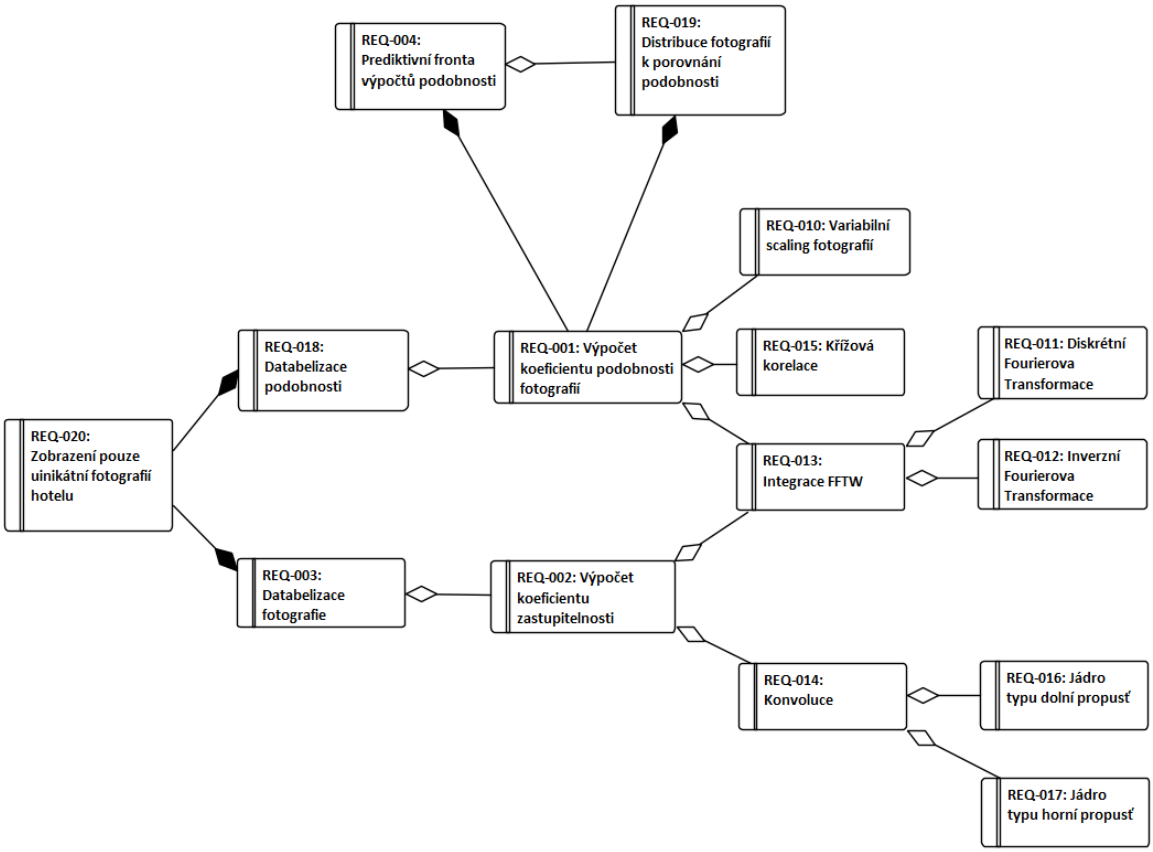


Obr. 7.2 L1 pohled cílového stavu

7.2 Požadavky

7.2.1 Funkční požadavky

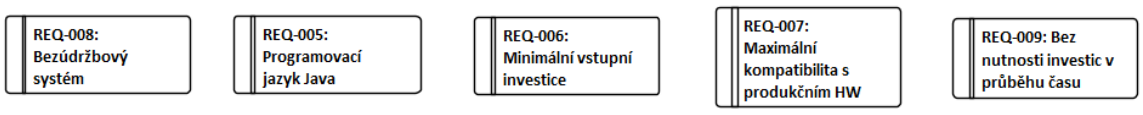
Klasifikaci funkčních požadavků systému znázorňuje obrázek (Obr. 7.3).



Obr. 7.3 Funkční požadavky

7.2.2 Nefunkční požadavky

Klasifikaci nefunkčních požadavků systému znázorňuje obrázek (Obr. 7.4).

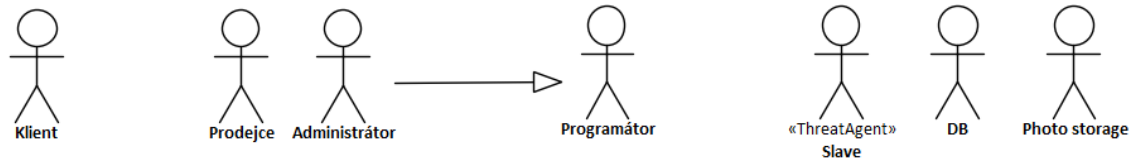


Obr. 7.4 Nefunkční požadavky

7.3 Případy použití

7.3.1 Aktéři

Klasifikaci aktérů (Obr. 7.5).



Obr. 7.5 Aktéři

7.3.2 Případy užití

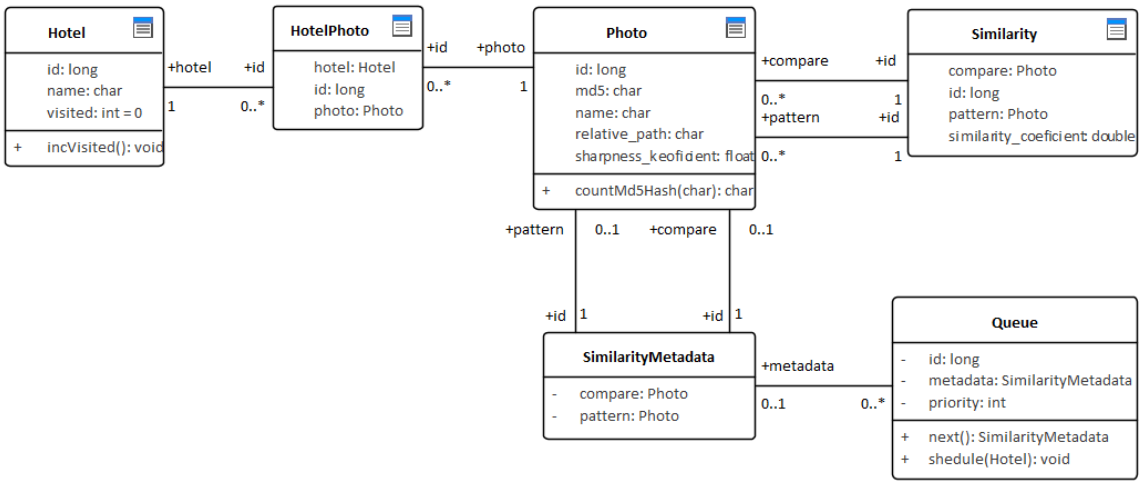
Klasifikaci případů použití znázorňuje obrázek (Obr. 7.6).

Obr. 7.6 Případy užití

7.4 Modely

7.4.1 Model tříd

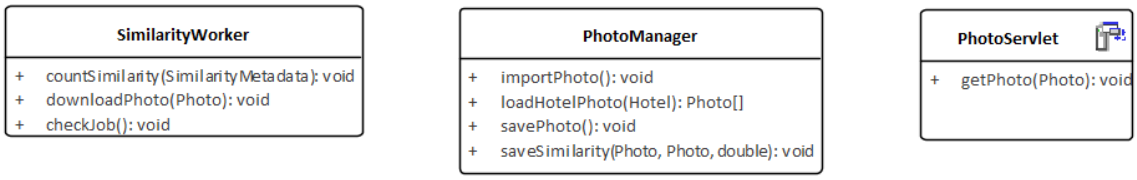
Identifikované datové modely jsou znázorněny pomocí modelu třídy na obrázku níže (Obr. 7.8)



Obr. 7.7 Model tříd

7.4.2 Model služeb

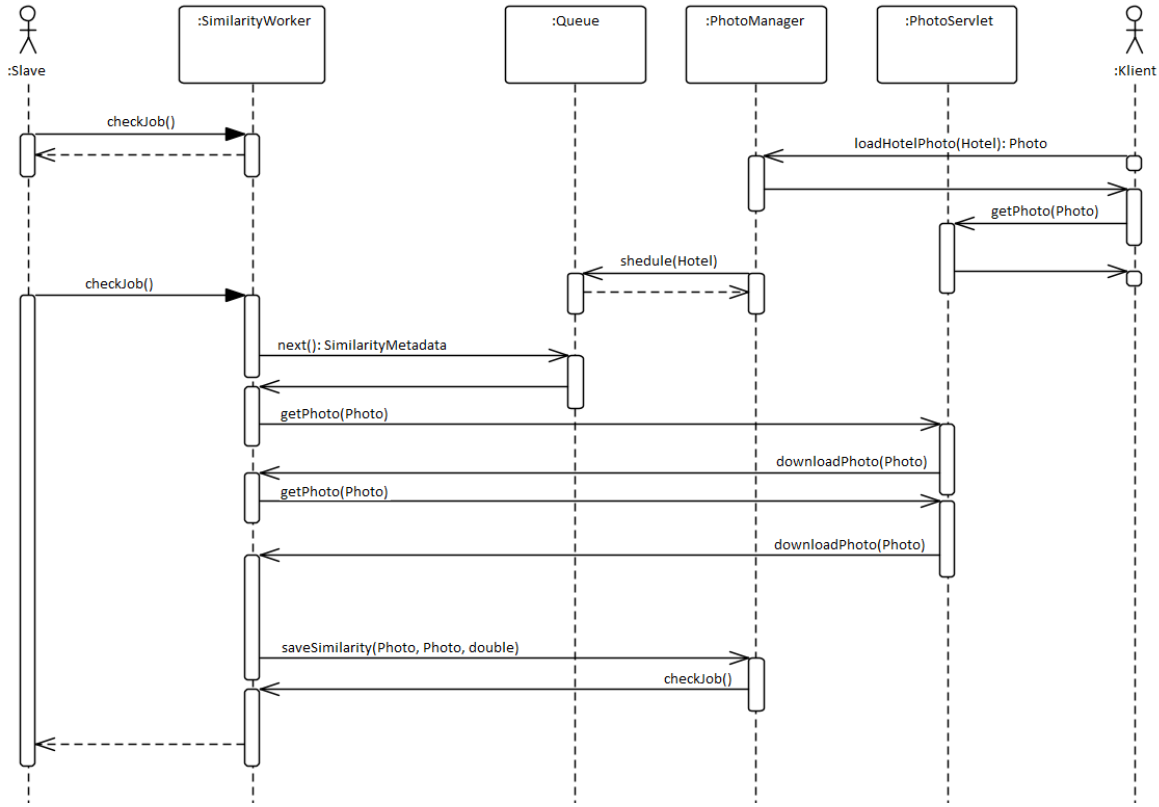
Identifikované backendové služby jsou znázorněny pomocí na obrázku níže (Obr. **??**)



Obr. 7.8 Model služeb

7.5 Sekvenční diagram distribuované služby

Na obrázku níže (Obr. 7.9) je znázorněn sekvenční diagram přibližující odbavení výpočtu KoF.



Obr. 7.9 Sekvenční diagram

8 Serverová (sběrná) strana distribuované služby

8.1 Databelizace fotografií

Proces určení KoZ je již trochu složitější. Nejprve bude představen zjednodušeně a následně rozvedeny jednotlivé kroky.

- Převod fotografie (Obr. 8.1a) na světlostní matici
- Převod světlostní matice do vlnového spektra pomocí DFT
- Sestavení matice ostrosti (matice hran)
  - Převod světlostní matice ve vlnovém spektru na matici hran (Obr. 8.1b),
  - za pomocí konvoluce typu horní propusť.
- Sestavení matice přeostřenosti, tj. matice hran za prahovou hodnotou (Obr. 8.1c)
  - Převod matice ostrosti ve vlnovém spektru na matici přeostřených hran,
  - za pomocí konvoluce typu horní propusť.
- Převod vypočtených matic zpět z vlnového spektra pomocí IFT.
- Sečtení hran v matici ostrosti a sečtení přeostřených hran v matici přeostřenosti.
- Podíl sumy hran a sumy přeostřených hran je KoZ.

Příklad pro představení si, jak tento proces funguje znázorňuje obrázek níže (Obr. 8.1). Po oddělení přeostřených hran (Obr. 8.1c), od všech hran (Obr. 8.1b) je již snadné vypočítat jejich poměrný koeficient.

Hlavní část určení KoZ je čistě experimentální, jelikož nebyl nalezen žádný srovnatelný vzorek či projekt, na kterém by šlo empiricky stavět.

Je nutné ručně nastavit do algoritmu některé konstanty (např. Negativní koeficient konvolučního jádra pro horní propusť) a jejich správnost následně testovat. Výsledkem je neutrální podklad z kterého "vystupují" hrany, velké množství hran $\Leftrightarrow$ fotka není rozmazaná.

1. horní propusť - obaz hran 2. dolní propusť na obrazu hran - tím docílíme odstraněním velmi ostrých hran (přeostřenosti) – tím spočítám počet ostrých hran Obrázek

8.2 Fronta nezpracovaných obrázků

8.3 Servlet pro stažení obrázků

8.4 Odeslání výsledků

(a) Fotografie před úpravou

(b) Vizualizace hran

(c) Vizualizace přeostřených hran

Obr. 8.1 Příklad přeostřenosti fotografie

9 Klientská (výkonná) strana distribuované služby

ZÁVĚR

Text závěru

f

SEZNAM POUŽITÉ LITERATURY

[1] ECKEL, Bruce. *Thinking in Java.* 4th ed. Upper Saddle River, NJ: Prentice Hall, c2006. ISBN 01-318-7248-6..

[2] WALLS, Craig. *Spring in action.* Fourth edition. Texas: Manning, 2013. ISBN 978-161-7291-203.

[3] HUNT, Charlie a Binu JOHN. *Java performance.* Upper Saddle River, NJ: Addison-Wesley, c2012. Java series. ISBN 01-371-4252-8.

[4] PERŮTKA, Karel. *MATLAB - Základy pro studenty automatizace a informačních technologií.* Zlín: Univerzita Tomáše Bati ve Zlíně, 2005. ISBN 80-731-8355-2.

[5] DOBEŠ, Michal. *Zpracování obrazu a algoritmy v C#.* Praha: BEN - technická literatura, 2008. ISBN 978-80-7300-233-6.

[6] MD5. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2018-05-08]. Dostupné z: `https://en.wikipedia.org/wiki/MD5`

[7] *Computer vision* [online]. [cit. 2018-05-08]. Dostupný z: `http://midas.uamt.feec.vutbr.cz/ZVS/Exercise02/content_cz.php`.

[8] Pixel. *In: Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2018-05-08]. Dostupné z: `https://cs.wikipedia.org/wiki/Pixel`

[9] RGB. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2018-05-08]. Dostupné z: `https://cs.wikipedia.org/wiki/RGB`

[10] *IEEE Transactions on Computers* [online]. 1972, **C-21**(2), 179–186 [cit. 2018-05-08]. DOI: 10.1109/TC.1972.5008923. ISSN 0018-9340. Dostupné z: `http://ieeexplore.ieee.org/document/5008923/`

[11] Korelace. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2018-05-09]. Dostupné z: `https://cs.wikipedia.org/wiki/Korelace`

[12] Cross-correlation. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2018-05-09]. Dostupné z: `https://en.wikipedia.org/wiki/Cross-correlation`

[13] *FFTW* [online]. [cit. 2018-05-08]. Dostupný z: `http://www.fftw.org`.

[14] The Design and Implementation of FFTW3. In: *Proceedings of the IEEE* [online]. 2005, **93**(2), s. 216–231 [cit. 2018-05-08]. DOI: 10.1109/JPROC.2004.840301. ISSN 0018-9219. Dostupné z: `http://ieeexplore.ieee.org/document/1386650`

[15] *IEEE Transactions on Image Processing* [online]. **5**(8), 1266-1271 [cit. 2018-05-08]. DOI: 10.1109/83.506761. ISSN 10577149. Dostupné z: `http://ieeexplore.ieee.org/document/506761/`

[16] *Shared Scientific Toolbox in Java* [online]. [cit. 2018-05-08]. Dostupné z: `http://freshmeat.sourceforge.net/projects/shared`

[17] *CMake* [online]. [cit. 2018-05-08]. Dostupný z: `https://cmake.org`.

[18] Wrapper. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2018-05-09]. Dostupné z: `https://en.wikipedia.org/wiki/Wrapper`

[19] In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2018-05-10]. Dostupné z: `https://cs.wikipedia.org/wiki/Konvoluce`

[20] Detekce hran. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2018-05-10]. Dostupné z: `https://cs.wikipedia.org/wiki/Detekce_hran`

[21] SOAP. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2018-05-10]. Dostupné z: `https://en.wikipedia.org/wiki/SOAP`

[22] Centrální procesorová jednotka. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2018-05-10]. Dostupné z: `https://cs.wikipedia.org/wiki/Centr%C3%A1ln%C3%AD_procesorov%C3%A1_jednotka`

[23] Graphic processing unit. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2018-05-10]. Dostupné z: `https://en.wikipedia.org/wiki/Graphics_processing_unit`

[24] Java (programovací jazyk). In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2018-05-10]. Dostupné z: `https://cs.wikipedia.org/wiki/Java_(programovac%C3%AD_jazyk)`

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

CPU	Central Processing Unit
DFT	Discrete Fourier Transform (Diskrétní Fourierova transformace)
FFTW	Fastest Fourier Transform in the West
GPU	Graphic Processing Unit
HW	Hardware
IFT	Inverse Fourier Transform (Zpětná Fourierova transformace)
KoP	Koeficient podobnosti dvou fotografií
KoZ	Koeficient zastupitelnosti vzájemně si podobných fotografií
MD5	Message-Digest algorithm
PC	Personal Computer
px	picture element (obrazový prvek)
SOAP	Simple Object Access Protocol
SW	Software

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

SEZNAM PŘÍLOH

PŘÍLOHA P I. NÁZEV PŘÍLOHY

Obsah přílohy